UNIVERSAL WIRELESS PROGRAMMER

EDL Project

S-01 Tue-16

FEEDBACK

Reviewer: Ankur Sir
& Maheshwar Sir

Feedback #1

Problem Decomposition: Instead of directly implementing custom RF, the problem should be broken down into subproblems with by Bluetooth (HC-05) as a proof of concept

Solutions Implemented

Bluetooth for Proof of Concept:

- A basic communication framework using Bluetooth will be developed to validate initial functionality before transitioning to more advanced solutions

RF IC Integration:

- The RF based implementation will optimize to provide measurable improvements over Bluetooth followed by Custom RF to replicate and enhance the IC for better customization

Feedback #2

Programming Approach: Initially, only development boards were considered, but the focus needed to shift to directly programming the Atmega328p chip using ISP while still using USB for development board programming.

Solutions Implemented

- Phase 1: USB-based programming of Arduino UNO board for initial testing and development.
- Phase 2: Direct programming of the Atmega328p chip using ISP through zif board to move beyond development boards.
- Phase 3: Expanding the solution towards a more universal programming approach by including other boards from Adruino family

MANAGEMENT PLAN

Overall Management

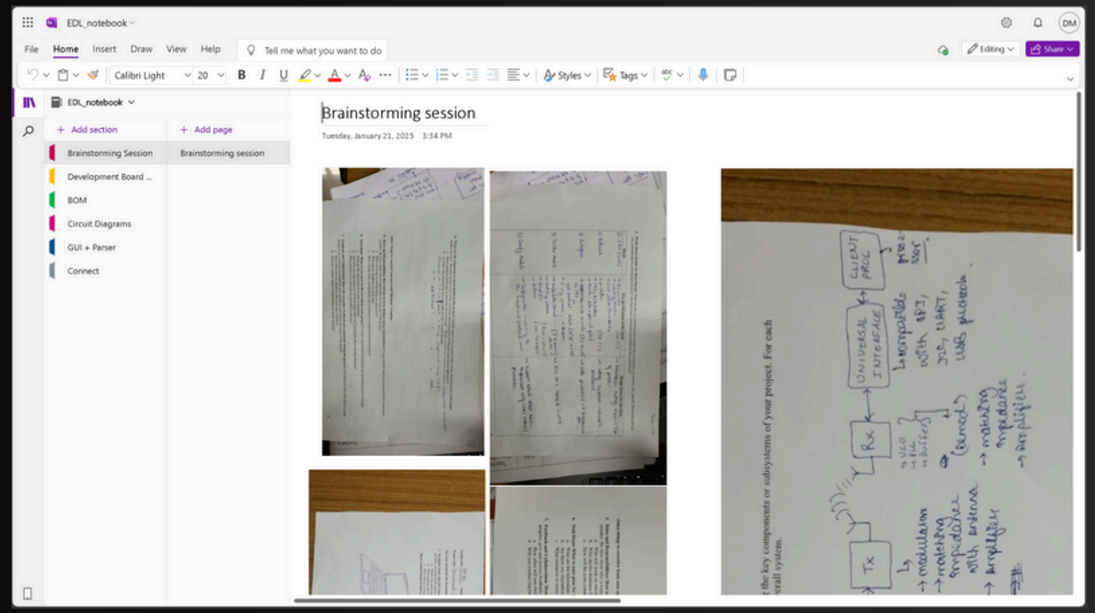
- For Milestone 2, the team was structured into three key focus areas to ensure an efficient division of tasks among members
  - Python GUI Development :Kaustav and Saptarshi
  - STM Receiver Implementation: Kaustav, Saptarshi and Dhruv
  - Board Detection and USB Protocol Development: Anay, Saptarshi and Hardik

- To ensure efficient collaboration, we organized task-specific sections in our OneNote notebook for systematic documentation

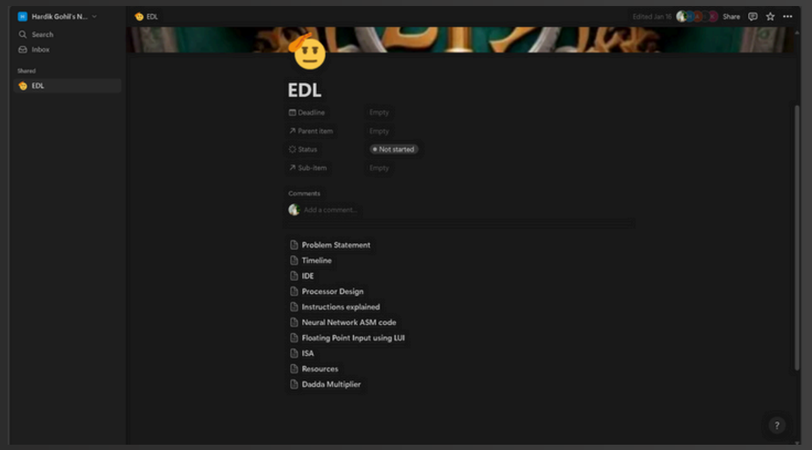
- Given the interdependencies among teams, we established clear deadlines for each subtask to maintain workflow continuity.

- Since multiple tasks required STM32 programming, we optimized development efforts by working in parallel with two different STM boards.

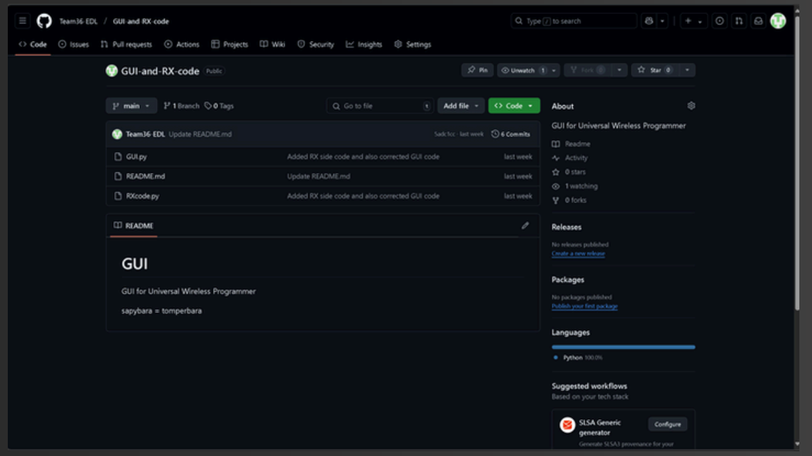
DOCUMENTATION



OneNote Notebook



Notion Page



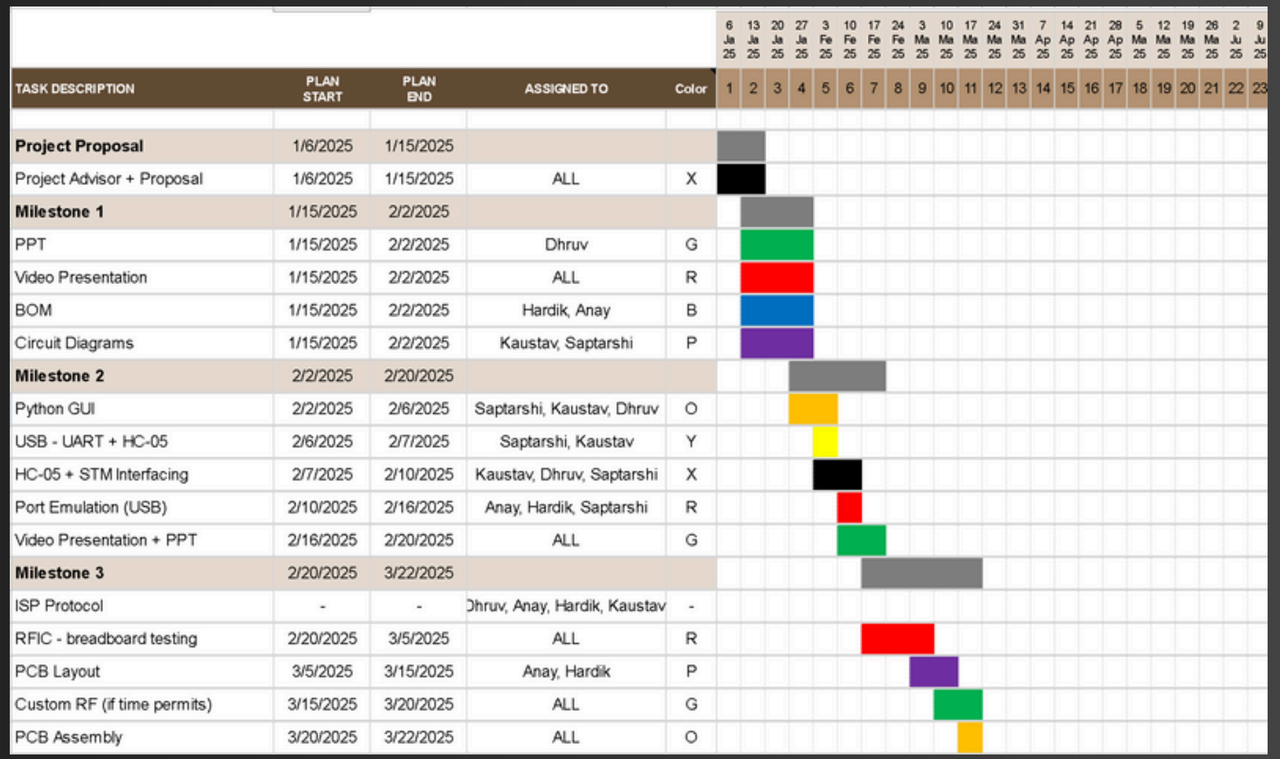
GitHub Repositories

GANTT CHART PROGRESS

- Due to significant changes in BOM after consulting Ankur and Maheshwar we could not meet the milestone 1 deadline. Our initial gantt chart is more or less invalid in view of this.
- We added an updated timeline for review 1 and have updated the Gantt chart with that information.
- We promised to perform the following major tasks in milestone 2 according to the updated timeline:
  - HC05-Transmission verification
  - python GUI for transmission of hex files
  - Port emulation
  - Programming Arduino over USB

- The only deviation from this is that we are facing some unprecedented issues in uploading hex code to Arduino. Specifically, triggering reset of Arduino by a command over USB from STM.

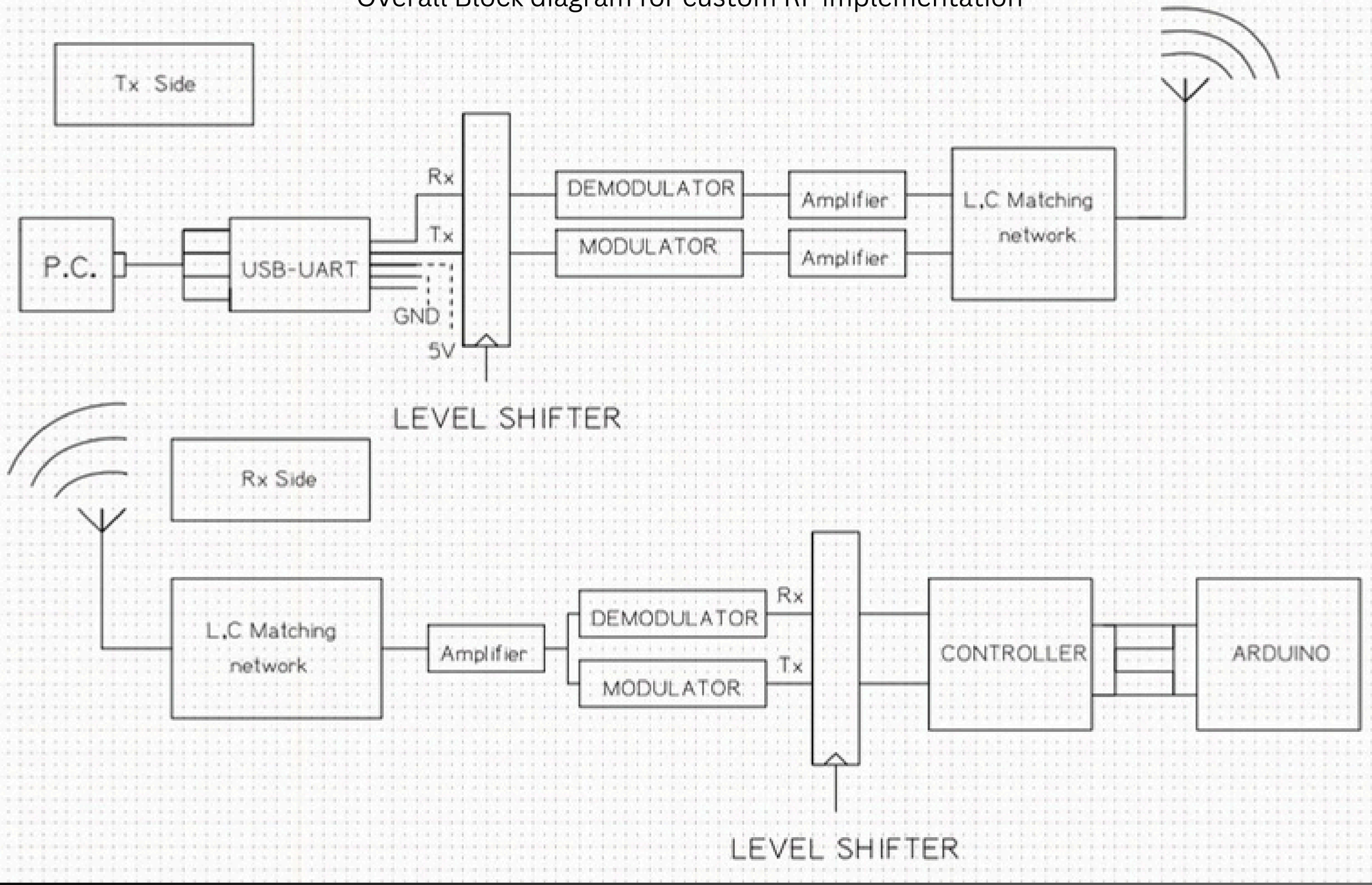
TASK DESCRIPTION	PLAN START	PLAN END	ASSIGNED TO	Color
Project Proposal	1/6/2025	1/15/2025		
Project Advisor + Proposal	1/6/2025	1/15/2025	ALL	X
Milestone 1	1/15/2025	2/2/2025		
PPT	1/15/2025	2/2/2025	Dhruv	G
Video Presentation	1/15/2025	2/2/2025	ALL	R
BOM	1/15/2025	2/2/2025	Hardik, Anay	B
Circuit Diagrams	1/15/2025	2/2/2025	Kaustav, Saptarshi	P
Milestone 2	2/2/2025	2/20/2025		
Python GUI	2/2/2025	2/6/2025	Saptarshi, Kaustav, Dhruv	O
USB - UART + HC-05	2/6/2025	2/7/2025	Saptarshi, Kaustav	Y
HC-05 + STM Interfacing	2/7/2025	2/10/2025	Kaustav, Dhruv, Saptarshi	X
Port Emulation (USB)	2/10/2025	2/16/2025	Anay, Hardik, Saptarshi	R
Video Presentation + PPT	2/16/2025	2/20/2025	ALL	G
Milestone 3	2/20/2025	3/22/2025		
ISP Protocol	-	-	Dhruv, Anay, Hardik, Kaustav	-
RFIC - breadboard testing	2/20/2025	3/5/2025	ALL	R
PCB Layout	3/5/2025	3/15/2025	Anay, Hardik	P
Custom RF (if time permits)	3/15/2025	3/20/2025	ALL	G
PCB Assembly	3/20/2025	3/22/2025	ALL	O



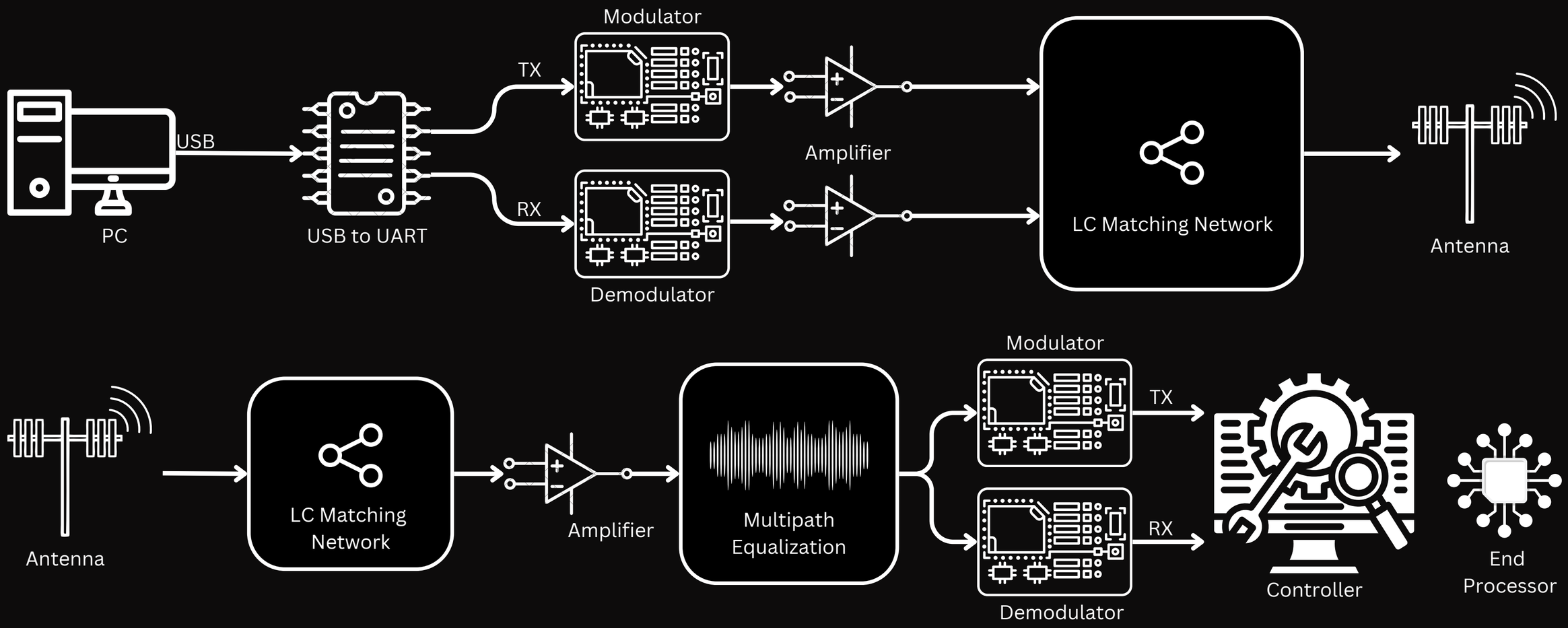
Updated Gantt Chart

BLOCK DIAGRAM

Overall Block diagram for custom RF implementation



PRINCIPLE OF OPERATION

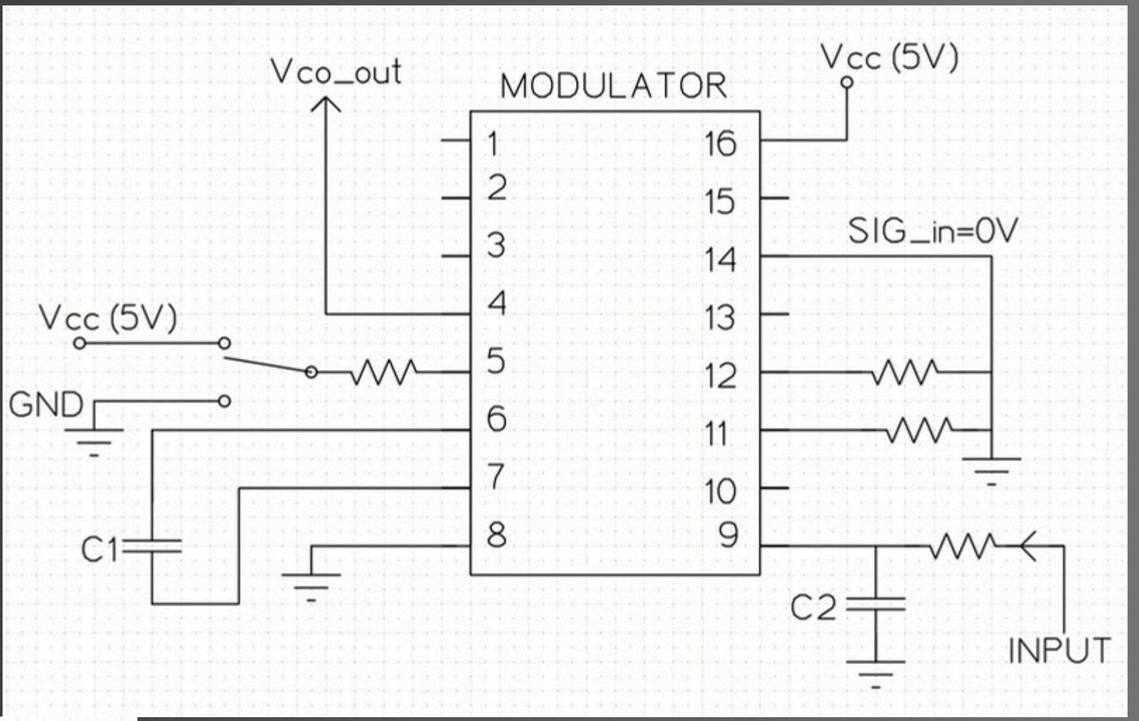


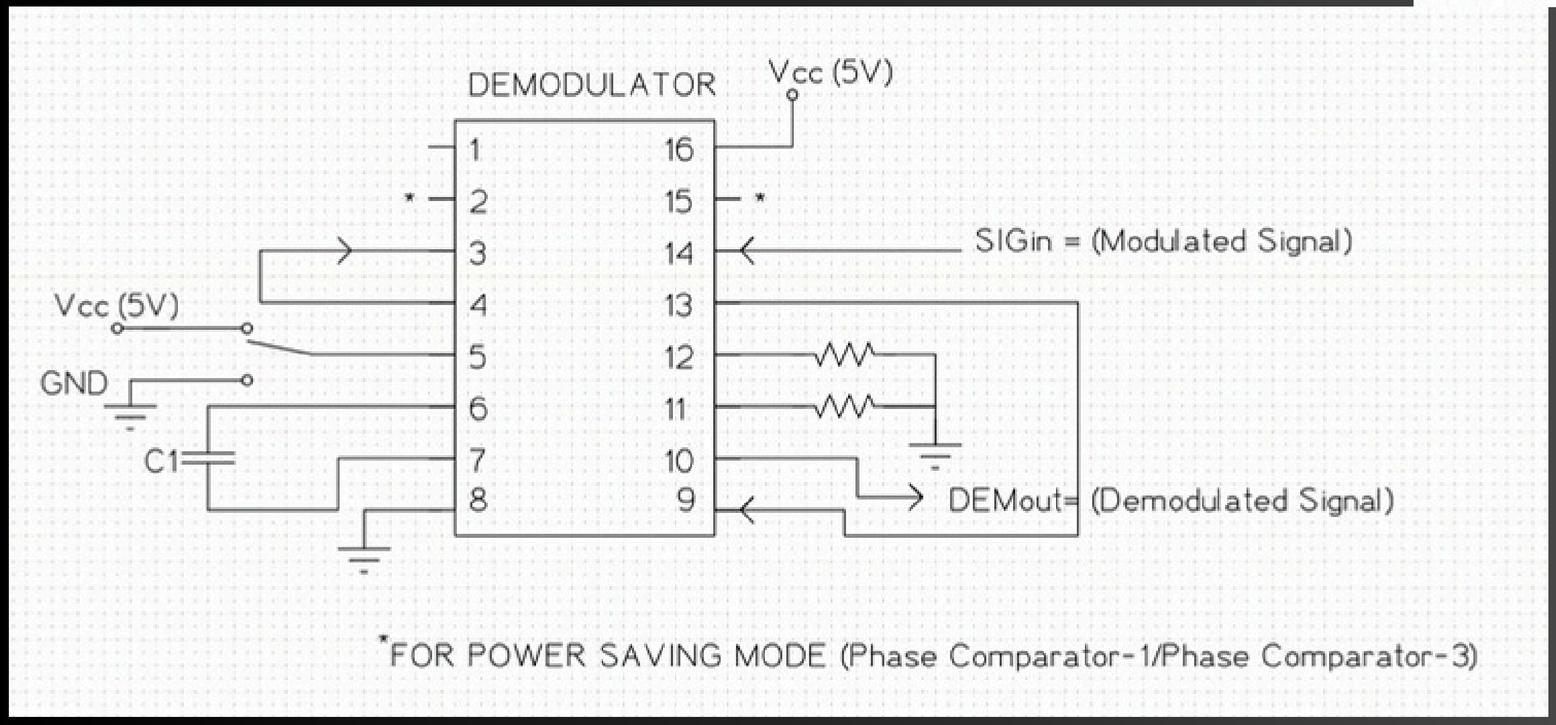
PRINCIPLE OF OPERATION (PC SIDE)

- The Python GUI sends the parsed hex file and the board selected.
- HC-05/SI464/Custom RF modules for tranmsitting the data
- Receiving the data on STM32, and matching with the actual signature read from the board/processor.
- If signature matches the hex file is uploaded, either using ISP or USB protocol depending on whether we have a processor IC or a development board at the end.
- For ISP protocol, hex file with bootloader needs to be selected in the GUI and for the USB protocol without bootloader is expected.

CIRCUIT DIAGRAMS

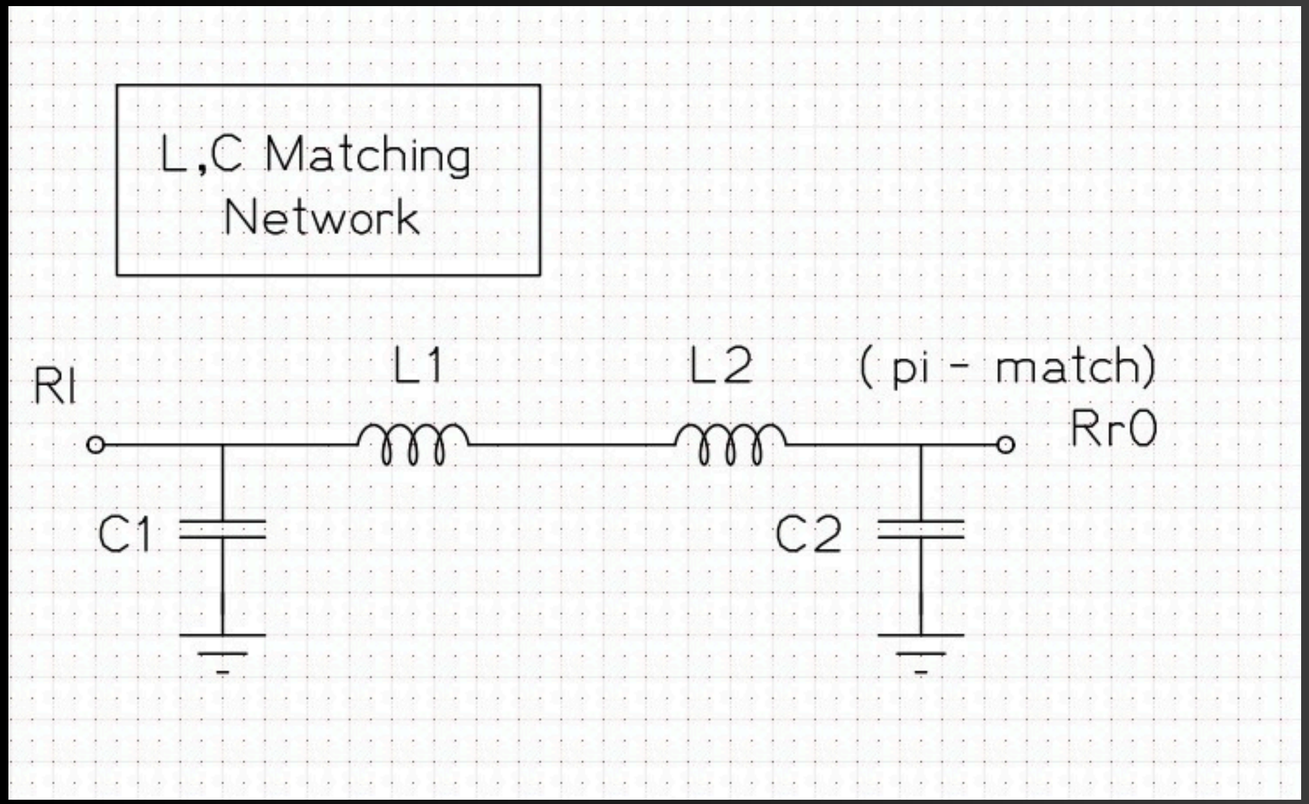
MODULATOR



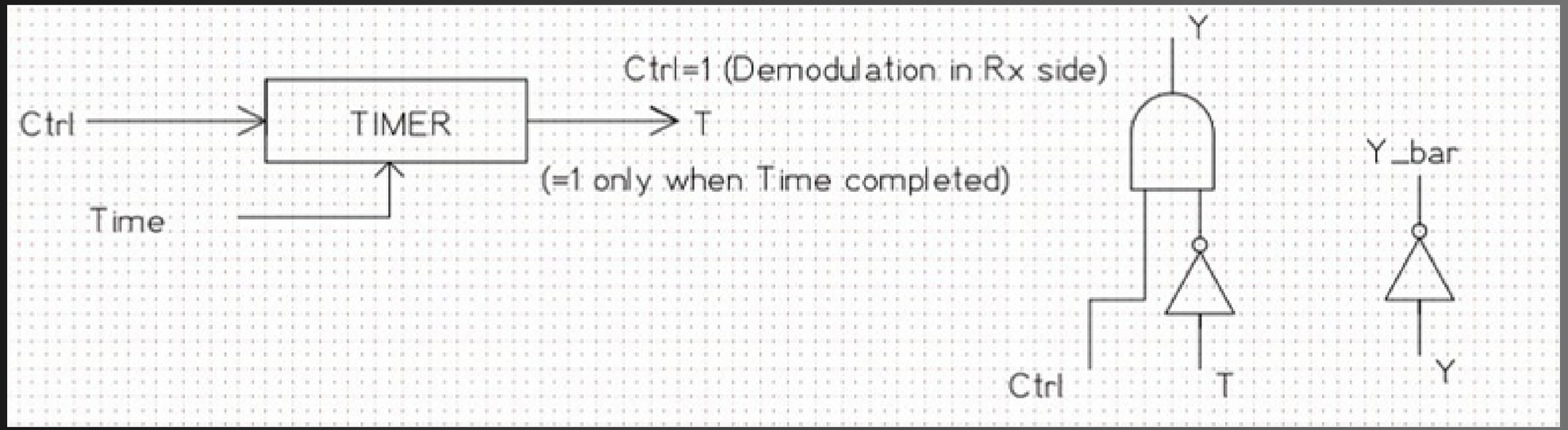


DEMODULATOR

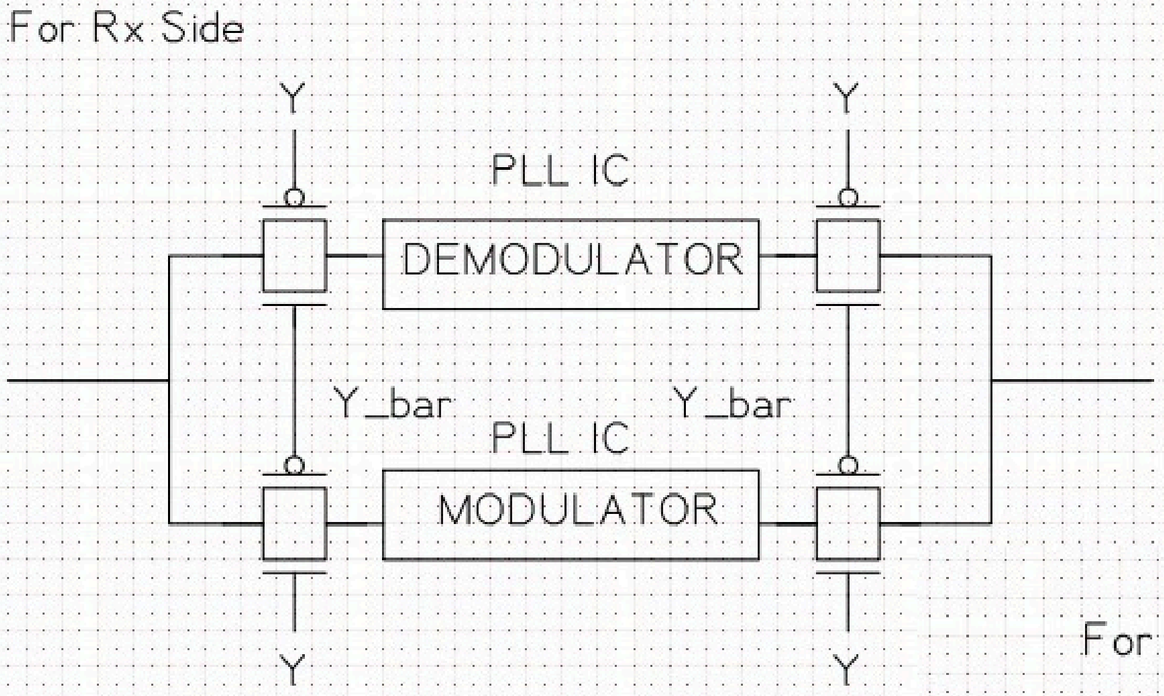
MATCHING NETWORK



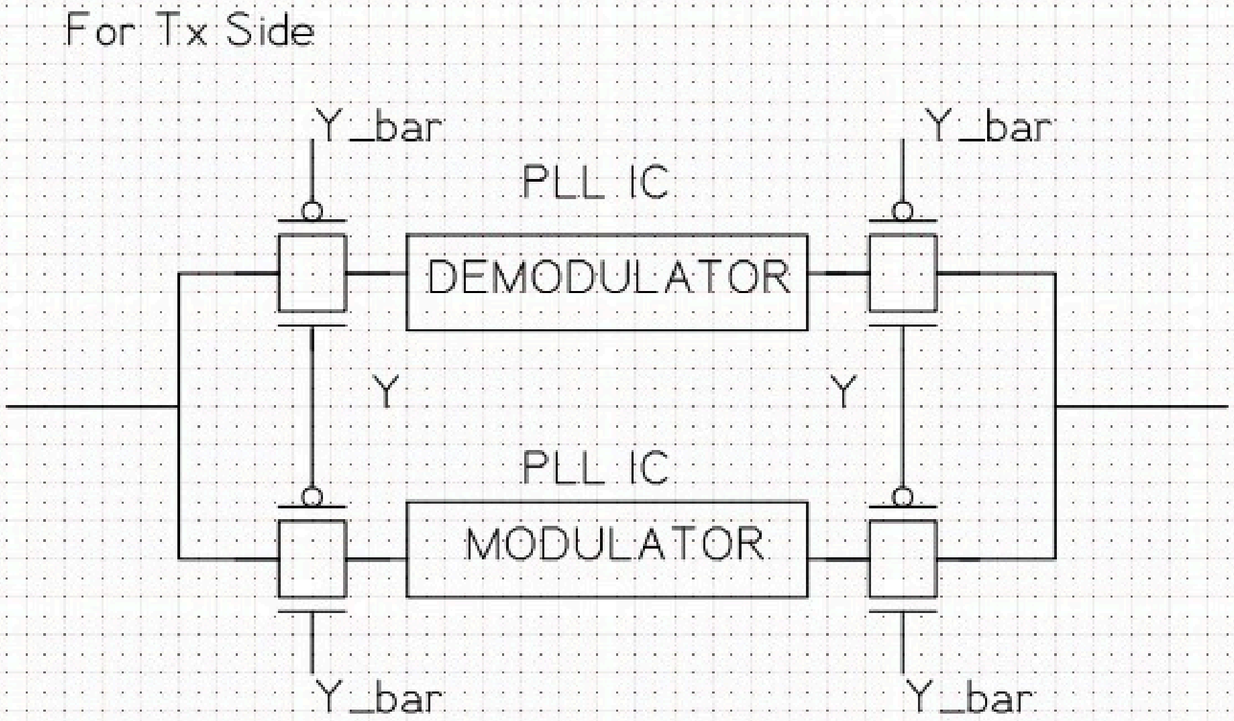
TIMING LOGIC

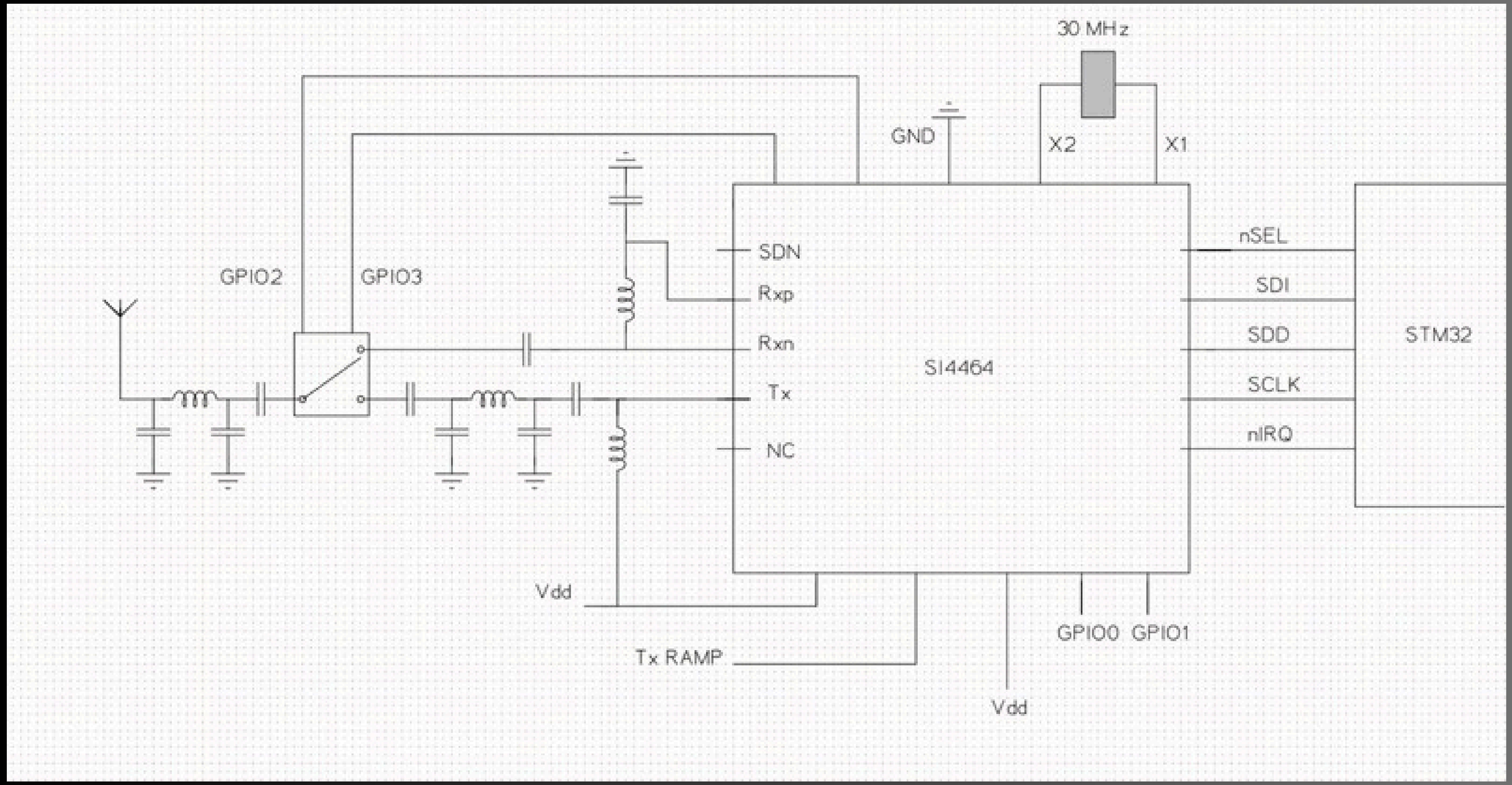


MODE SWITCHING LOGIC (RX)

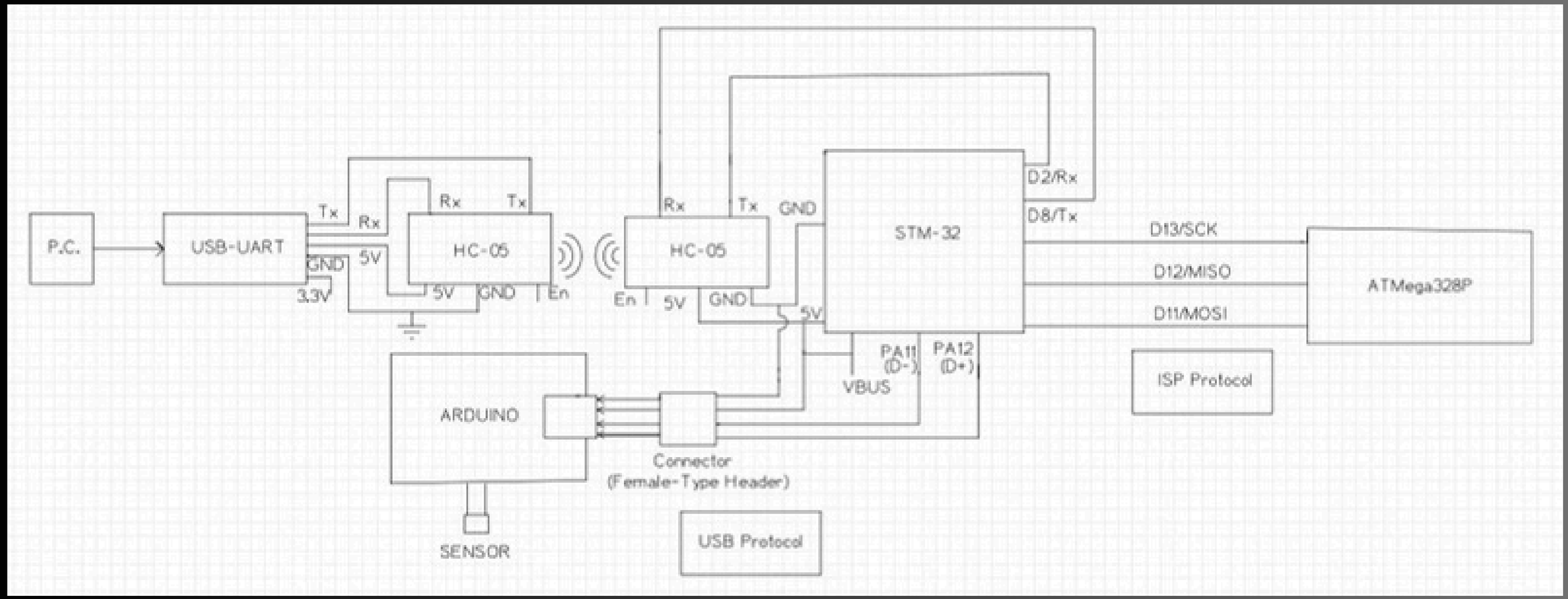


MODE SWITCHING LOGIC (TX)




30 MHz
GND
X2
X1
GPIO2
GPIO3
SDN
Rxp
Rxn
Tx
NC
SI4464
nSEL
SDI
SDD
SCLK
nIRQ
STM32
Vdd
Tx RAMP
Vdd
GPIO0
GPIO1

SI 4464

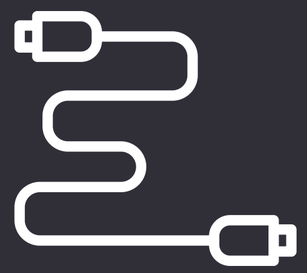


OVERALL CIRCUIT (ISP+USB PROTOCOLS)

ANALYSIS

STM Receiver Implementation (using HC-05)

- Using ATMODE in both the HC-05 modules set them as master and slaves
- Bounded the master to the address of the slave, so that it doesn’t connect to other available receivers
- Once the WT module is connected to the PC, the GUI detects the port automatically and connects over bluetooth to the receiver STM32 board



GUI - STM - HC05

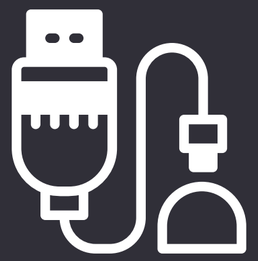
- Once signature is matched, upload the whole hex file from PC to receiver STM 32 board
- Load the desired hex file from the GUI, select the board.
- Firstly, upload the signature and ensure signature match

ANALYSIS

Board Detection and USB Protocol Development

USB Enumeration Process

- The code continuously checks for connected USB devices and logs the status, such as detecting if the connected device matches a predefined selected board.
- Identifies the connected USB device using its Vendor ID and Product ID (PID/VID) for accurate classification (e.g., Arduino).
- Once a device is detected, the code can send commands (e.g., reset commands) to the device using a specific protocol (STK500).



Board Detection

- It verifies if the connected device matches the user-selected board and logs a mismatch if the devices are different.
- Used UART to defer print statements to realterm.
- Used STM in OTG host mode and capable of supporting various types of USB devices eg: Communication Device Class, MTP class, HID class, etc.

BOARD DETECTION

- Program Starts:
  - main() → HAL_Init() → SystemClock_Config() → MX_GPIO_Init() → MX_USART2_UART_Init() → MX_USB_HOST_Init()
- Main Loop:
  - Inside while (1), check_usb_device() is called → USBH_DevDescTypeDef → classify_usb_device()
  - Check if device is detected → If Detected, compare with selected_board_name using strcmp()
  - If matched, print success, else print mismatch.

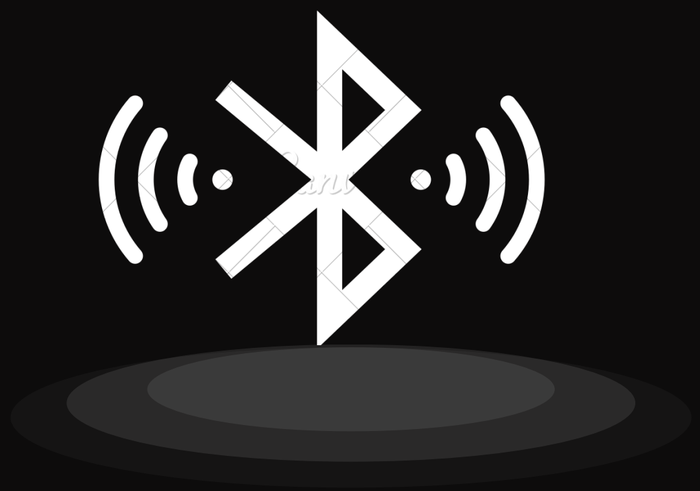
- USB Host Processing:
  - MX_USB_HOST_Process(): Continuously processes USB communication tasks in the background.

BOOTLOADER MODE

- If Arduino is detected:
  - Check if USB CDC is ready ->If ready, call reset_arduino().
  - reset_arduino() toggles DTR to reset the device.
- The DTR pin from the USB-to-Serial chip (ATmega16U2) is connected to the RESET pin of the ATmega328P microcontroller through a 100nF capacitor.
  - When DTR goes LOW, it creates a pulse that temporarily pulls the RESET pin LOW, resetting the microcontroller.
  - send_sync_request() sends STK500 sync request (0x30 0x20).->Wait for response->If 0x14 0x10 is received, bootloader mode is confirmed.
  - If a sync request is received, the bootloader allows new firmware to be uploaded.

TESTING

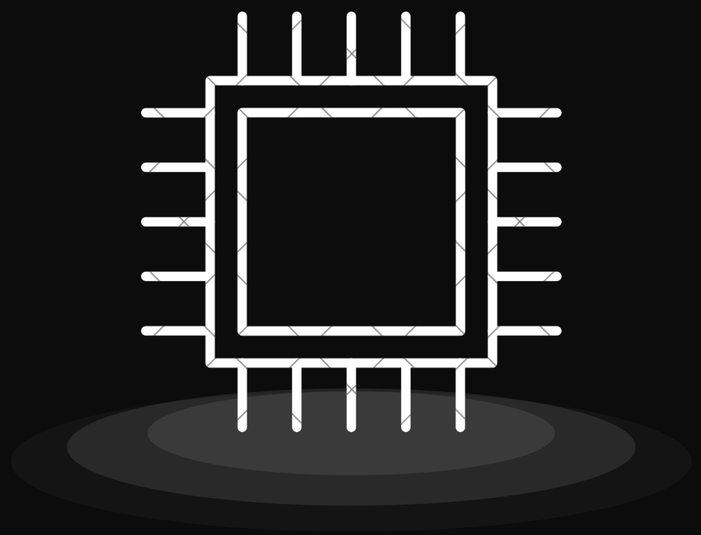
PHASE 1



Bluetooth

- Using HC05 for wireless transmission over bluetooth
- Simple since it supports configuration over UART

PHASE 2



RF IC

- Using SI4464 for programming wirelessly
- Supports much longer range than Bluetooth
- Configured over SPI protocol

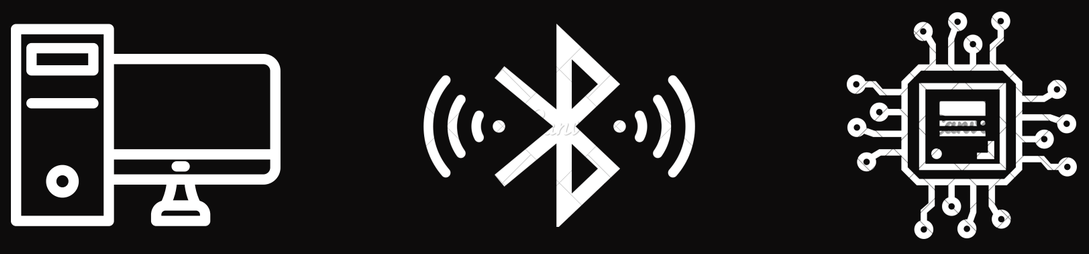
PHASE 3

Custom RF

- Implementing RF from scratch using modulation and demodulation techniques
- Adaptive Frequency control

TESTING PHASE 1

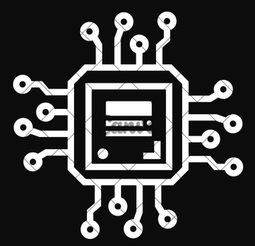
PART 1

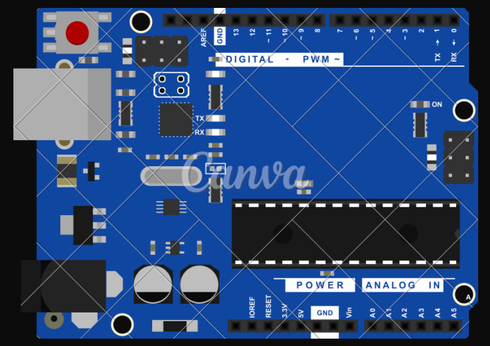


PC ⟶ WT ⟶ WR

- This takes care of making sure that the laptop can communiacate seamlessly with the Wireless Receiver over HC05 using bluetooth
- It also takes care of creating a GUI and sending appropriate commands to the WR over a wireless channel

PART 2



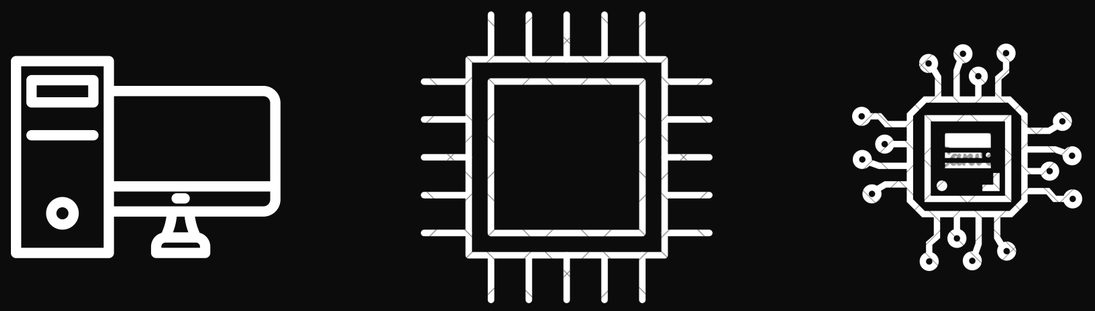


WR ⟶ Development Board

- This part is about implementing the handling of the USB protocols on the MCU
- Making it capable of communicating with a variety of end processors which includes development boards as well as microcontroller chips

TESTING PHASE 2

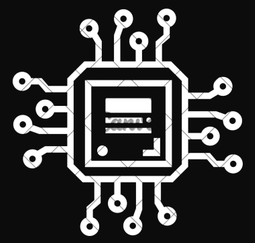
PART 1

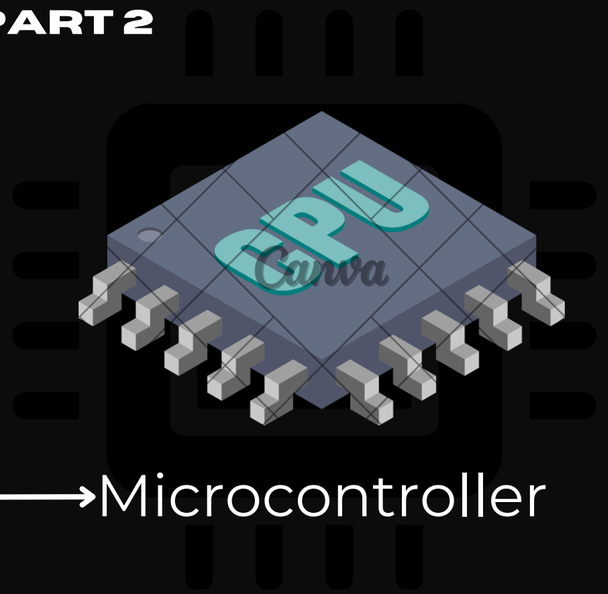


PC ⟶ WT ⟶ WR

- Now we will move on to interfacing with RF IC SI4464 which has much better range
- This IC supports programming via SPI so we will need to also interface SPI to UART converters and vice versa

PART 2





WR ⟶ Microcontroller

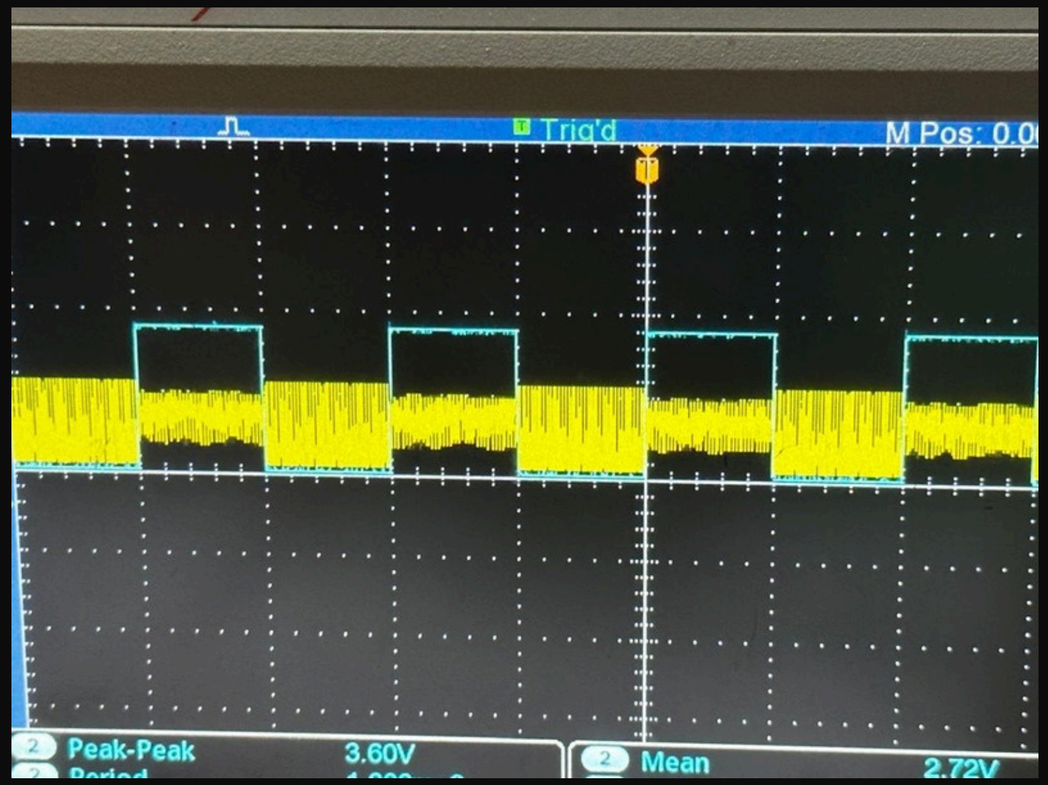
- This part is about implementing the programming to microcontroller ICs
- In this stage, we will add support to ISP protocol for programming the flash memory of the chip

TESTING PHASE 3

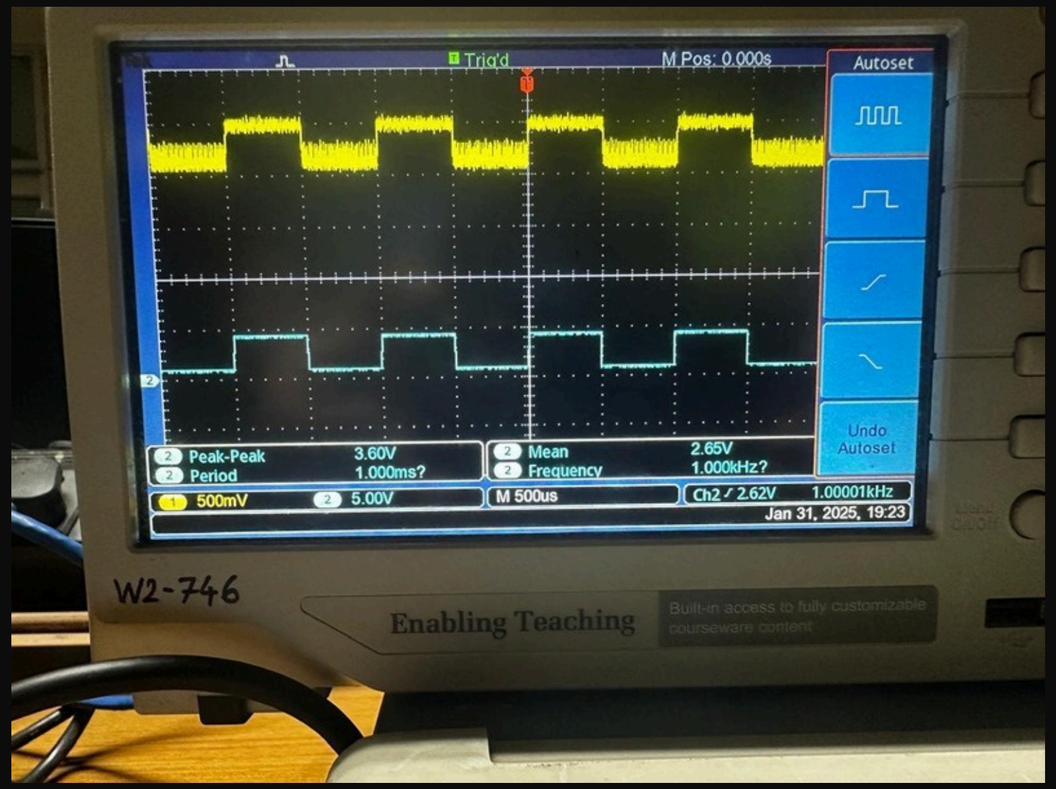
- Test Modulation/Demodulation via Frequency Shift Keying over a wired network
- Matching Network Testing
- Interfacing between bytes received/sent and the input/output of the communication network
- Controlling bidirectional nature of the channel
- Controlling frequency and range of communication
- Hamming code implementation

TESTING RESULTS

FSK OVER WIRED MEDIUM

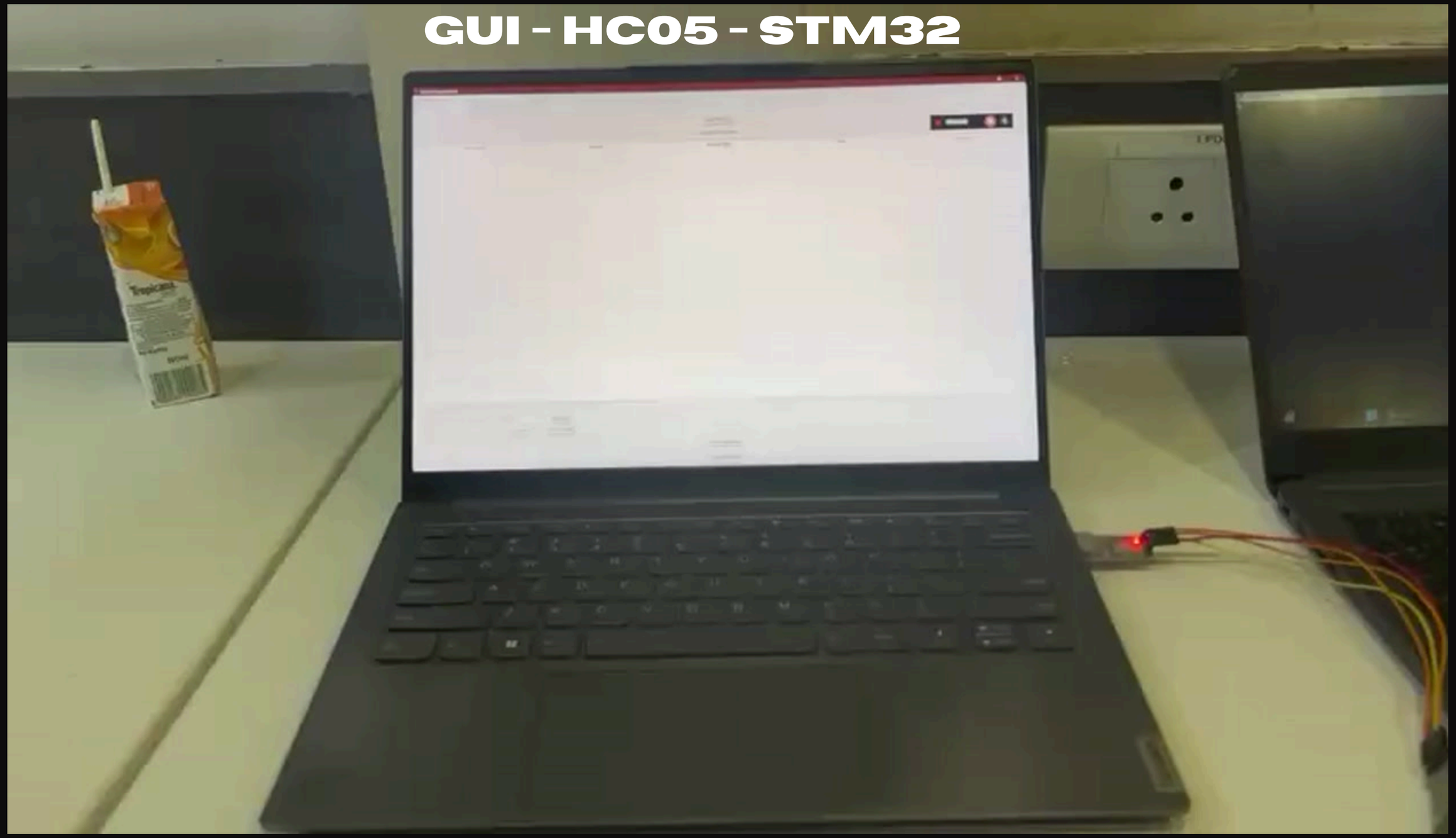


Modulated Signal



Demodulated Signal

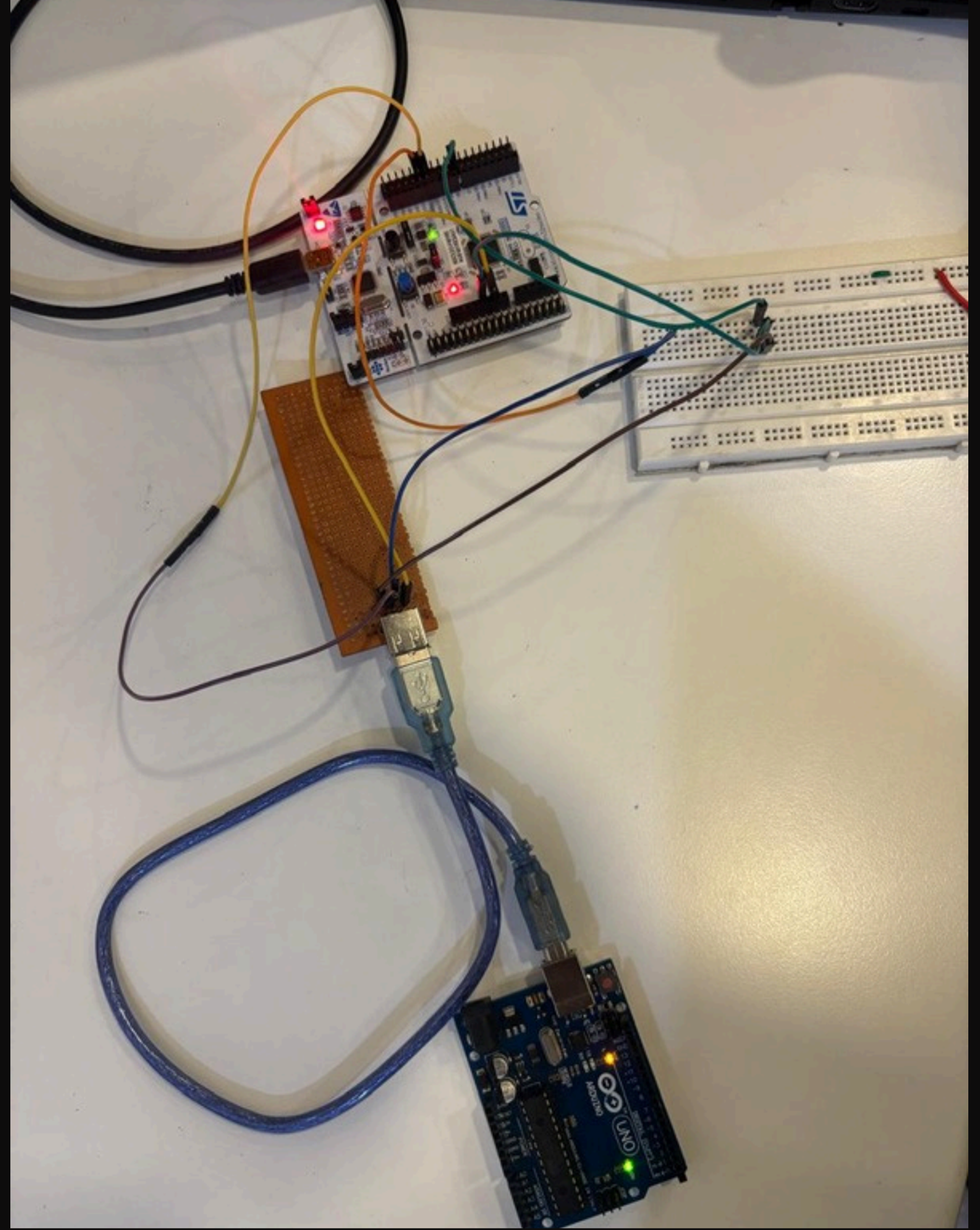
GUI - HC05 - STM32



STM32 - ARDUINO

STM - END PROCESSOR

Using female A type connector



THINKING AHEAD

- Interfacing the bootloader on development board with our hex file for successful upload of the code
- ISP protocol Implementation
- Replacing the HC-05 modules with SI 4464 and testing the received data using checksum
- Testing the SI 4464 on breadboard and then implement it on a PCB
- Expand to more processors of the ATMega family
- Expand GUI into Universal IDE so that serial plotting/monitoring of sensor reading is also made feasible

THANK YOU